PC 클러스터를 이용한 실시간 분산 웹 영상 내용기반 검색 시스템에 관한 연구

이은애[†] · 하석운[††]

요 약

최근의 내용기반 영상 검색 시스템은 한정된 수의 영상을 저장해 놓은 단일의 서버를 이용하고 있다. 이로 인해 웹 상의 다양한 영상을 원하는 웹 사용자의 요구를 만족시키지 못하고 있다. 수많은 웹 영상을 대상으로 하는 내용기반 영상 검색 시스템은 무엇보다도 실시간에 기반을 두어야 한다. 이를 구현하기 위해서는 영상 수집과 특징 추출에 걸리는 많은 소모 시간 문제가 해결되어야 한다. 최근, 고속의 데이터 처리를 목적으로 부하분산 PC클러스터가 개발되고 있다. 본 논문에서는 많은 시간을 요하는 영상 수집과 특징 추출 작업을 부하분산 PC 클러스터의 종속 컴퓨터들에 분배함으로써 전체 검색 시간을 감소시켰으며, 이를 통해 실시간 웹 영상 검색의 가능성을 발견할 수 있었다.

A Study on the Real-time Distributed Content-based Web Image Retrieval System using PC Cluster

Lee Eun-Ae[†] and Ha Seok-Wun[††]

ABSTRACT

Recent content-based image retrieval systems make use of a local single server contained a limited number of images. So these systems are not satisfactory for the Web user's needs that make request for various images on the Web. A content-based image retrieval system that has regard for a great number of Web images has to stand on the basis of real-time first of all. Therefore, to implement the above system we have to resolve a problem of large waste time to take for an image collection and feature extractions. In recent, PC clusters with a load distribution are implemented for the purpose of high-performance data processing. In this paper, we decreased the whole retrieval time by distributing the tasks of image collection and feature extraction to take much time among the slave computers of the PC cluster, and so we found the possibility of the real-time processing in the retrieval of Web images.

1. 서 론

반도체 기술의 발전으로 CPU의 처리 속도는 지난 수십년간 10^3배 정도의 개선을 이루었고 광통신 기술의 발전으로 통신망의 속도는 10^5배 이상으로 개선되었다[1].

그러나, 웹 사용자의 증가는 지난 몇 년간 10^8배에 이르고 웹 문서에 포함되어 있는 이미지 수의 증가는 웹 사이트 증가율의 몇십배가 넘는다.

웹 사용자나 웹을 이용한 정보 처리의 증가율이 하드웨어 기술의 발달 속도에 비해 결코 뒤지지 않는 것이다. 따라서 여전히 CPU의 처리 속도와 통신망의 속도는 소프트웨어 개발의 걸림돌이 되고 있다.

물론, 처리 시간적 중요도가 높은 핵 발전과 프로세스 제어 등과 같은 응용 분야에 사용되는 컴퓨터 시스템에서 수행되는 많은 태스크들은 분산 처리되어 지고 엄격한 실시간 제한을 갖고 있으며 그에 관

[†] 준회원, 경상대학교 강사
[††] 정회원, 경상대학교 컴퓨터과학과 교수, 컴퓨터 정보통신 연구원

련한 연구들이 여러 방면으로 이루어지고 있다[2].

기존의 웹 영상 검색 분야에서는 대부분 서버에 저장된 영상 데이터베이스를 이용한 영상검색을 지원하고 있다[3]. 그러나 한정된 수의 영상으로는 웹 사용자의 요구를 만족시키기 어렵다. 따라서 다양하고 수많은 영상을 웹 사이트로부터 직접 제공할 필요가 있다. 그러나 웹 문서에 포함되어 있는 이미지의 수가 그 수를 짐작할 수도 없을 정도로 많은데다 이미지의 종류나 크기, 저장된 위치도 다 제각각이기 때문에 실시간으로 각 사이트에 접속하여 검색을 하기에는 너무나 많은 처리 시간이 요구되어 근본적인 제약이 따른다.

따라서 본 논문에서는 실시간 웹 영상 검색을 위해 PC 클러스터를 이용한 분산 처리를 행하고자 한다. 동일한 성능과 사양을 가진 PC들을 서로 연결하여 그 중 하나를 마스터 노드로 하고 나머지는 슬레이브 노드로 하여 마스터 노드의 지시하에 각각의 슬레이브 노드가 독립적으로 웹 사이트에 접속하여 이미지를 저장하고 저장한 이미지의 특징을 추출하여 마스터에 넘기면 마스터에서 유사 영상 검색을 하도록 하여 전체적인 처리시간을 최대한 줄임으로써 실시간 영상 검색이 이루어지도록 하였다.

2. PC 클러스터와 영상검색에 관한 고찰

2.1 PC 클러스터

CPU의 성능이 아무리 개선된다하여도 실제적인 시뮬레이션에 필요한 계산 용량을 지원해 줄 수 없으므로 이에 대한 대응책으로 다수의 저가 PC를 병렬로 연결하여 계산을 분할하여 실행시켜서 대단위 계산을 수행하려는 연구가 시도되었다. 클러스트링을 위한 API 프로그램으로는 범용의 병렬 환경을 제공하는 PVM(Parallel Virtual Machines)과 MPI(Message Passing Interface)가 발표되었고, 계속 개선되고 있다[4].

여러대의 컴퓨터를 서로 연결하여 마치 하나의 컴퓨터처럼 사용하는 것을 클러스터라고 하며 클러스터로 연결된 컴퓨터들은 서로 통신이 가능하며 업무를 분담해서 수행하게 된다.

PC 클러스터의 목적은 저가의 PC를 다수 연결하여 슈퍼컴퓨터의 성능을 초과하는 계산 장비를 구현하는 것이다. 현재의 기술로 슈퍼컴퓨터 가격의 1~2%의 자금으로 동등한 성능을 실현할 수 있다.

초창기의 클러스트링은 주로 페일오버에 중점을 두었지만 현재는 이외에도 동일한 하나의 작업을 클러스터 내에서 여러 대의 호스트로 분산시키는 로드밸런싱에 관심이 모아지고 있다.

클러스터는 크게 부하분산 클러스터와 고가용성 클러스터의 두 부류로 분류된다[5,6].

부하분산 클러스터는 다수의 컴퓨터가 하나의 프로그램을 협동으로 수행하여 고성능의 계산능력을 낼 수 있어 특정 분야에 있어서는 기존 슈퍼컴퓨터보다 가격대 성능비가 우수하다는 장점이 있고 web, ftp, database 등과 같은 서비스를 제공하기 위한 클러스터로써 서비스를 요청받으면 자신이 그 요청을 처리하지 않고 클러스터내의 다른 컴퓨터에게 서비스를 제공하도록 함으로써 대규모 서비스를 제공해 줄 수 있다.

고가용성 클러스터는 여러대의 컴퓨터를 가지고 있으며 그중 일부가 사용불능이 되었을 때 이를 극복하기 위함이 그 목적이다.

이를 위해서 서로가 서로를 수시로 체크하게되며 고장난 컴퓨터를 클러스터에서 동적으로 제거하고 해당 컴퓨터가 수행하던 특정 작업을 다른 컴퓨터가 대신 수행해 주기도 한다. 또한 다시 클러스터로 합류할수도 있다.

낮은 가격과 높은 가용성을 유지하면서 증가하는 네크워크 요청에 효과적으로 대응하기 위한 해결책으로 여러대의 호스트를 묶어 네트워크 서비스를 제공하는 클러스트링 기술이 각광받고 있으며 계속 연구되고 있다[7].

본 연구에서는 웹 영상 검색 과정 중 가장 많은 처리 시간을 요하는 영상의 수집 및 특징 추출을 부
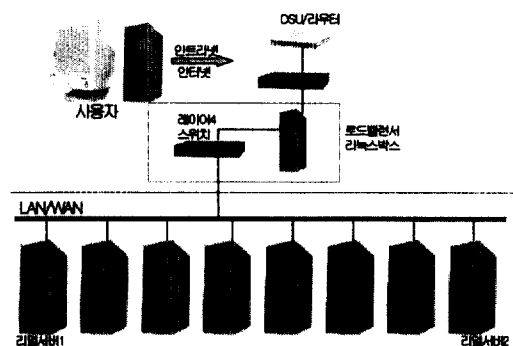
그림 1. 웹/인터넷 클러스터링

하분산 클러스터를 이용해 처리하고자 한다.

웹 영상을 수집할 때 네트웍의 상태에 따라 처리 속도가 많은 영향을 받는다. 검색 대상이 되는 영상의 URL이 평균 300개라고 가정하면 하나의 컴퓨터로 처리하고자 했을 때, 실제 실험의 경우(본 연구에서 사용한 컴퓨터는 펜티엄-IV, 1.4GHz, 256M RAM 이었다) 네트웍의 흐름이 원활하고 영상의 크기가 대체적으로 5KB 이하인 작은 영상의 경우에는 100여개, 네트웍의 흐름은 원활한데 영상의 크기가 대용량인 영상이 섞여 있는 경우에는 50개 이하, 네트웍 트래픽이 혼잡한 경우라면 20개도 간신히 수집할 수 있었다. 물론 컴퓨터 cpu의 처리속도를 높이고, 메모리를 확장하여 수집했을 때 수집 가능한 영상 수도 늘어났지만 10개에서 20여개 가량의 차이가 있을 뿐이었다. 즉, 컴퓨터 사양의 확장만으로는 해결할 수가 없었다. 따라서 본 연구에서는 그 해결책으로 부하분산 PC 클러스터를 이용하여 위의 문제를 해결해 보고자 한다. 고가의 슈퍼컴퓨터대신 저렴한 PC들을 묶어 처리율을 높이고자 하는 것이다.

본 연구에서 사용한 PC 클러스터는 모두 4대로 구성되어 있다. 물론, 더 많은 컴퓨터를 클러스터로 연결하면 더 좋은 결과를 얻을 수 있겠지만 연구 환경의 제약으로 4대로 구성된 PC 클러스터를 사용하여 실험을 하였다.

2.2 영상 검색

영상을 검색하는 방법에는 텍스트기반 검색과 내용기반 검색이 있다. 텍스트 기반 검색은 영상의 제목이나 영상의 특징 등을 텍스트로 저장하여 검색하는 방법이고[8], 내용기반 검색은 영상의 특징인 색상, 질감, 모양, 공간적 관계에 근거하여 검색하는 방법이다[8-10].

색상에 근거하는 방법으로는 도미넌트 컬러를 이용하는 방법과 컬러 히스토그램을 이용하는 방법, 색상의 이름을 이용하는 방법, 컬러-유도 감각에 대한 방법이 있으며[10] 질감에 근거하는 방법으로는 공간기반 모델에 의한 방법, 주파수 기반 모델에 의한 방법, 텍스쳐 시그네쳐에 의한 방법이 있다[10]. 또한, 모양에 근거한 방법으로는 영상 내에서 색상이나 밝기가 급격히 변화하는 부분을 에지라고 하는데 이러한 에지의 방향 성분들을 조사함으로써 물체의 질감이나 모양, 위치의 유사성에 대해서 판단할 수 있고, 특징기반 방법과 모양 변형 기반 방법도 있다[10,11].

본 논문에서는 두 단계의 검색과정을 거치는데 첫 번째로는 먼저, 사용자의 질의어를 대상으로 텍스트 기반 검색을 하여 후보영상들을 수집하고, 두 번째로는 후보영상들 중 하나의 질의영상을 대상으로 내용기반 검색을 하도록 하였다. 또한 내용기반 검색을 위해서는 영상의 색상에 근거하는 컬러 히스토그램을 적용하여 검색하였다.

3. 실시간 분산 웹 영상 검색 시스템의 구현

본 장에서는 PC 클러스터를 이용한 내용기반의 실시간 분산 웹 영상 검색 시스템(이하 RDWIRS : Real-time Distributed Web Image Retrieval System)을 설계하고 구현한다.

3.1 RDWIRS의 설계

RDWIRS의 전체 시스템 구성은 그림 2와 같으며, 크게 사용자 인터페이스, 웹 문서 분석 및 영상검색, 영상 수집 및 특징 추출의 세 부분으로 구성되어 있다. 그 중에서 가장 많은 시간이 소모되는 영상 수집 및 특징 추출 부분을 3개의 슬레이브 노드들이 동시에 실행하도록 하여 전체적인 처리시간을 단축시키는 데 중점을 두었다.

본 연구에서 사용한 PC 클러스터는 동일한 성능을 갖는 4대의 PC를 연결한 부하분산 클러스터 시스템을 사용하였으며 PC 클러스터의 마스터 컴퓨터는 사용자 인터페이스와 웹 문서 분석 및 영상검색을 담당하며, 3개의 슬레이브 컴퓨터는 영상 수집 및 특징 추출을 담당한다.

이러한 PC 클러스터 내부의 정보 흐름을 세부적으로 나타내면 그림 3과 같다. 마스터 노드에서는 영
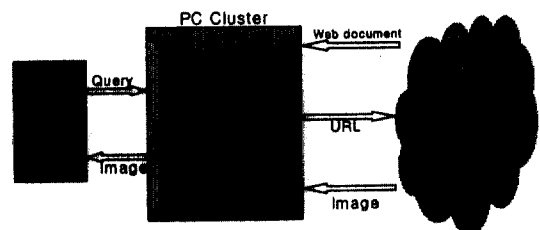
그림 2. RDWIRS의 구조
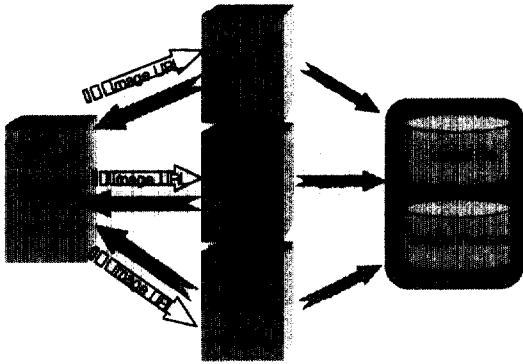
그림 3. PC 클러스터의 처리 구조

상의 URL을 각각의 슬레이브 노드에 전달 해 주고 슬레이브 노드들은 다른 슬레이브 노드와는 상관없이 독자적으로 영상 URL에 접속하여 영상을 획득하고 특징 추출을 한 다음, 영상 파일과 영상 특징 파일을 디스크에 저장하는 일을 한다. 모든 영상 URL에 대해 동일한 처리를 한 후 모든 작업이 끝나면 제어를 마스터 노드에 반환한다.

RDWIRS의 전체 처리과정은 그림 4의 흐름에 따른다.
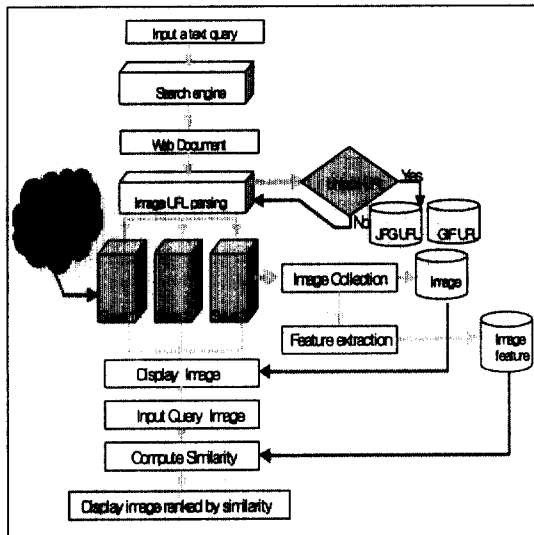
그림 4. RDWIRS의 전체 처리 과정

사용자 인터페이스는 사용자의 질의를 받고 검색되어진 결과를 출력해주는 일을 한다. 본 시스템의 사용자 인터페이스는 마스터 노드에서 운용되는 것으로 사용자가 웹으로 접근할 수 있도록 하였다.

웹 문서 분석 및 영상 검색은 PC 클러스터의 마스터 노드가 담당하며, 이 마스터 노드는 사용자의 입력을 처리하여 관련 있는 웹 문서를 구한 뒤, 웹 문서로부터 영상의 URL을 추출하여 슬레이브 노드에 넘기고, 슬레이브 노드에서 획득한 영상을 사용자 인터페이스에 출력 해 주는 일을 한다. 그런 다음, 사용자가 클릭한 영상을 질의 영상으로 하여 검색 대상 영상들과의 유사도를 계산한 후 우선 순위별로 유사영상을 출력해 주는 일을 한다.

그리고, 영상 수집 및 특징 추출은 PC 클러스터에서 3개의 슬레이브 노드들이 담당하며 이 슬레이브 노드들은 마스터 노드로부터 받은 영상 URL을 이용하여 웹으로부터 직접 영상을 획득하고 획득된 영상의 특징을 추출하는 일을 한다.

사용자의 입력을 처리하는 PC 클러스터의 작업을 세분화하면 웹 문서 분석 및 영상을 검색하는 일을 하는 마스터 노드는 그림 5와 같은 처리 과정을 갖는다.

슬레이브 노드의 처리 과정은 그림 6과 같으며
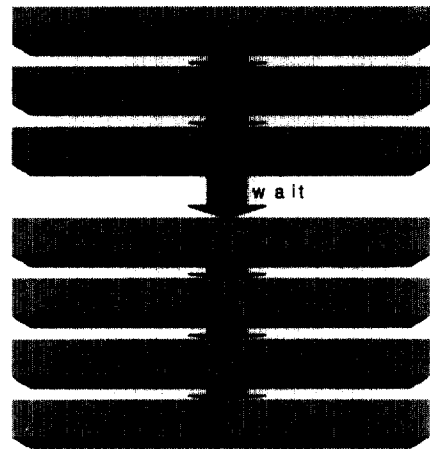
그림 5. 마스터 노드의 처리 과정
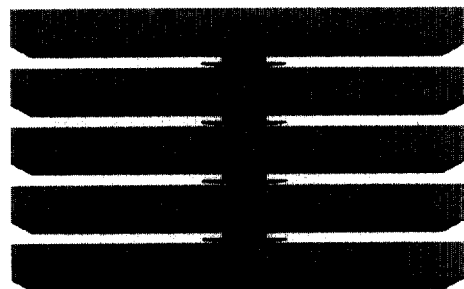
그림 6. 슬레이브 노드의 처리 과정

영상 수집 및 특징 추출은 실시간 분산 웹 영상 검색에서 가장 많은 처리시간이 필요한 부분이기 때문에 여러개의 슬레이브 노드에서 동시에 영상을 수집하도록 하였다.

3.2 RDWIRS의 구현

3.2.1 텍스트 기반 질의

사용자 인터페이스는 웹에서 접근이 가능하도록 HTML 문서로 만들었고 검색하고자 하는 영상의 이름을 간단히 텍스트 형태로 입력할 수 있도록 하였다.

사용자 인터페이스를 통해 입력된 텍스트 기반 질의를 위한 검색어는 표 1과 같은 과정을 통해 질의를 추출한 후 이미지 파일 검색 엔진의 질의어로 사용한다.

본 연구에서는 검색 엔진 "엠파스(http://search.empas.com/search/image_search.html)" 에서 제공하는 "이미지 파일 검색" 엔진을 활용하여 텍스트 기반 검색을 하였다. 이미지 파일 검색 엔진에서는 질의어를 입력하면 입력한 질의어를 포함한 영상의 URL을 검색해 웹 문서 형태로 출력 해주는 일을 한다. 텍스트 기반 질의 처리 과정은 그림 8과 같다.
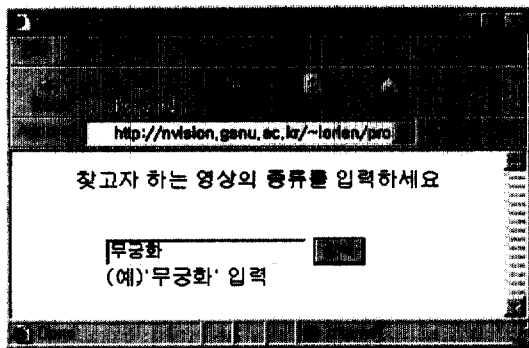
그림 7. 초기 사용자 인터페이스

표 1. 질의어 추출 알고리즘

```
get_query(char *query){
    int cl, i=0;
    char buffer[256];
    FILE *qf;

    cl=atoi(getenv("CONTENT_LENGTH"));
    fread(buffer,cl, 1, stdin);
    buffer[cl]='\0';
    while(buffer[i++]!='q');
    --i;
    while(buffer[i]!='&') *query++=buffer[i++];
}
```
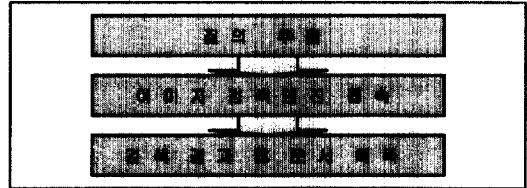
그림 8. 텍스트 기반 질의 처리 과정

3.2.2 웹 문서 분석 및 영상 URL 분배

웹 문서 안에는 다양한 형태의 HTML 태그들이 들어있다. 그 중에서 <a>을 찾아 검색된 영상 URL의 목록이 들어있는 .HTML 파일 리스트에 대한 링크목록 파일을 그림 9와 같이 만들고, <img>를 찾아 파일 종류별로 영상 URL 파일(GIF목록 파일, JPEG 목록 파일)을 그림 10과 같이 만든다.

또한, 링크목록 파일로부터 같은 과정을 반복하여 최종적인 영상 URL 파일을 완성한다.

각 영상의 URL 목록 파일은 영상 URL의 중복을 체크하여 유일한 URL에 대해서만 저장한다.
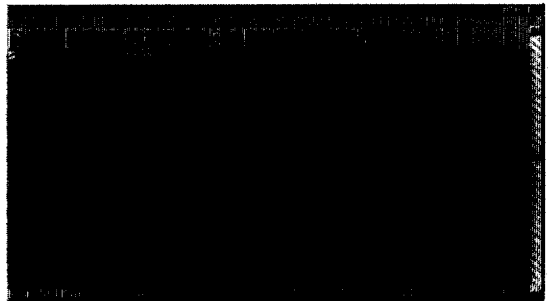
그림 9. 링크 목록 파일
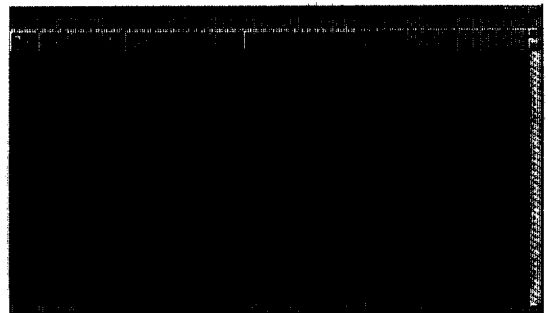
그림 10. GIF 목록 파일

마스터 노드는 영상의 URL 목록 파일이 완성되면 표 2와 같은 영상 분배 알고리즘을 적용하여 슬레이브 노드들에 그 분배 목록을 넘긴다.

슬레이브 노드들은 각각 n 개(전체 영상 URL의 수/슬레이브노드의 수)씩의 영상을 처리한다.

표 2. 영상 분배 알고리즘

```
MPI_Status Stat;
 MPI_Init(&argc,&argv);
 MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
 MPI_Comm_rank(MPI_COMM_WORLD,&myid);

 if(myid==0){
   for(i=1;i<numprocs;i++){
     sbuf=i;
     MPI_Send(&sbuf,1,MPI_INT,i,tag1,MPI_COMM_
WORLD);
   }
   for(i=1;i<numprocs;i++){
     MPI_Recv(imgfl,200*50,MPI_INT,i,tag2,MPI_
COMM_WORLD,&Stat);
   }
 }else{

MPI_Recv(&procs,1,MPI_INT,0,tag1,MPI_COMM_
WORLD,&Stat);
     for(j=procs-1;j<numimgs;j+=numprocs){
       get_img(imgURL[j]);
     }
     MPI_Send(imgflist,200*50,MPI_INT,0,tag2,MPI_
COMM_WORLD);
}
```

3.2.3 영상 수집 및 특징 추출

슬레이브 노드들은 각자 서로 다른 영상 URL을 기반으로 영상을 수집하고 또한 그 영상의 특징을 추출하는 일을 한다.

영상을 수집하기 위해서는 HTTP/1.0 프로토콜을 사용한다. HTTP 프로토콜은 요구/응답방식으로 동작하며 원하는 서비스를 요구(GET, POST, HEAD...)를 하면 데이터 송수신을 위한 TCP 연결이 만들어지고, 서버가 응답을 보내어 데이터 전송을 끝내면 자동적으로 연결이 끊어지게 된다. 표 3과 같은 C언어와 Unix 소켓을 이용한 루틴으로 GIF영상 및 JPEG 영상을 획득할 수 있다. 헤더가 포함된 GIF 영상 스트림과 JPEG 영상 스트림의 헤더를 제거한 후 .GIF 파일과 .JPEG 파일로 저장한다. 영상 스트림의 헤더 제거 루틴은 헤더와 GIF 스트림, 혹은 JPEG 스트림을 구분하는 기준 단어가 버퍼 메모리 크기 내에 고정적으로 존재하기 때문에 receive() 시스템 콜이 존재하는 루프 내에서 바로 행해지도록 한다. GIF 스트림과 JPEG 스트림의 헤더 제거 루틴을 표 4와 표 5에 각각 나타내었다.

표 3. 영상 스트림 획득 루틴

```
url_binding(char *purl){
   struct hostent *host;
   host=gethostbyname(purl);
   addr=((struct in_addr *)host->h_addr)->s_addr;
   svc_addr.sin_family=AF_INET;
   svc_addr.sin_addr.s_addr=addr;
   svc_addr.sin_port=htons(80);
}
sock_init(){
   sd=socket(AF_INET, SOCK_STREAM,0);
   connect(sd, (struct sockaddr *)&svc_addr,
            sizeof(struct sockaddr_in));
}
get_jpg(){
       ⋮
  url_binding(hname);
  sprintf(msg, "GET %s HTTP/1.0\n\n", jpgurl[loop]);
  si=sock_init();
  send(sd, msg, strlen(msg),0);
       ⋮
}
```

일반적으로 영상의 색상에 기반하는 검색에는 영상의 밝기 빈도수를 그래프로 나타낸 컬러 히스토그램을 사용한다. 컬러 히스토그램을 사용하여 영상의 특징을 추출하는 이유는 다른 특징 추출 방법에 비하여 처리 시간이 빠르기 때문이다.

즉, 웹 상의 영상은 글자를 이미지로 표현한 단순한 영상에서부터 실세계의 모습을 담은 컬러사진이나 화가들의 작품에 이르기까지 영상의 모습이 매우 다양하며 그 크기 또한 다양하다.

표 4. GIF 스트림 헤더 제거 루틴

```
while((n=recv(sd,buf, sizeof buf-1, 0))>0){
   j=0;
   if(flag==0){
       for(j=0; j<n; j++){
        if(buf[j]=='G'&& buf[j+1]=='I' &&
            buf[j+2]=='F'){
               flag=1;
               break;
          }
       }
   }
   for(i=j;i<n;i++)
       fprintf(fn,"%c",buf[i]);
}
```

표 5. JPEG 스트림 헤더 제거 루틴

```
while((n=recv(sd,buf,sizeof buf-1, 0))>0){
  j=0;
  if(flag==0){
      for(j=0; j<n; j++){
        if(buf[j]=='J'&&buf[j+1]=='F'&&
          buf[j+2]=='I'&&buf[j+3]=='F'){
            j=j-6;
            flag=1;
            break;
        }
      }
  }
  for(i=j;i<n;i++)
      fprintf(fn,"%c",buf[i]);
}
```

이러한 다양한 모습을 갖는 영상을 대상으로 한 검색에서 모양 특성이나 질감 특성을 적용하기에는 처리 시간이 지나치게 많이 걸리기 때문에 실시간 검색을 목적으로 하는 본 시스템에서는 다루지 않도록 하였다.

컬러 히스토그램은 256 밝기 레벨을 가진 R, G, B 채널 히스토그램을 연결하여 일련의 배열로 배치한 것을 말한다.

슬레이브 노드에서는 수집한 영상의 컬러 히스토그램을 표 6의 루틴으로 구하여 영상 특징 파일을 만든다. 컬러 히스토그램은 영상 파일을 다운로드 받은 다음 구하도록 작성되어 있으며, 여기서 GIF 영상은 256칼라로 구성되어 있기 때문에 별도의 파일 변환작업 없이 파일 헤더 정보를 이용하여 컬러 히스토그램을 구하고, JPEG 영상은 컬러 히스토그램을 구하기 위해 먼저 BMP 파일로 변환을 하여야 한다.

표 6. 컬러 히스토그램 계산 루틴

```
data=(unsigned char *)malloc(biWidth*biHeight);

 for(i=0;i<256;i++){
  for(j=0;j<4;j++){
   if(j==3) getc(fp);
   else{
    ch=getc(fp);
    pal[i][j]=ch & 0xff;
   }
  }
 }
 fread(data,1,biWidth*biHeight,fp);
 fclose(fp);
 //픽셀당 r,g,b 값 출력

 for(i=0;i<biHeight;i++)
  for(j=0;j<biWidth;j++){
   r=pal[*data][2];
   g=pal[*data][1];
   b=pal[*data++][0];
   fprintf(wfp, "\n%d %d %d", r, g, b);
  }
  fclose(wfp);
```

또한, 수집된 영상들의 크기가 다양하기 때문에 컬러 히스토그램을 구한 후 표준화작업이 필요한데 영상의 가로*세로 값을 구해진 컬러 히스토그램 값에 나누어 값의 범위를 0~1 사이의 값으로 표준화한 최종 특징 값을 저장한다.

3.2.4 내용기반 영상 검색

슬레이브 노드는 영상 수집과 특징 추출 작업이 끝나면 결과를 마스터 노드에 전송한다.

마스터 노드는 수집된 영상들을 사용자 인터페이스에 출력하고 사용자가 질의 영상을 선택하면 슬레이브 노드에서 구한 영상의 특징 값을 이용하여 사용자가 선택한 질의 영상과의 유사도를 계산한다.

질의 영상의 컬러 히스토그램 벡터를 H_Q 라 하고 비교 영상의 컬러 히스토그램 벡터를 H_P 라 하면 i 번째 검색 대상 영상과의 컬러 히스토그램 유사도 S_{Hi}는 식(1)과 같이 구할 수 있다. 유사도 계산에는 히스토그램 벡터 사이의 유클리드 거리(Euclidean Distance)를 적용하였다[9].

$$\begin{aligned} H_Q &= \{R_{Q_1}, .\, R_{Q_{256}}, G_{Q_1}, .\, .\, G_{Q_{256}}, B_{Q_1}, .\, .\, B_{Q_{256}}\} \\ H_P &= \{R_{P_1}, .\, R_{P_{256}}, G_{P_1}, .\, .\, G_{P_{256}}, B_{P_1}, .\, .\, B_{P_{256}}\} \\ d &= \sqrt{\textstyle\sum (H_P - H_Q)^2} \\ S_{H_i} &= 1 - d \end{aligned} \quad (1)$$

4. 실시간 분산 웹 영상 검색 결과 및 고찰

RDWIRS을 사용하여 검색어 "돌"과 "장미"를 입력하였을 때의 텍스트 기반 검색 결과를 그림 11과 그림 12에 나타내었다.

텍스트기반 검색을 시도하였기 때문에 영상의 URL에 "돌", "장미"라는 글자가 들어가거나 혹은 영상을 설명하는 HTML 텍스트에 "돌" , "장미"라는 글자가 들어간 모든 영상이 검색되었다.

검색어 "돌"을 입력하였을 때 검색된 영상 URL의 수는 모두 383개였고 실제 수집된 영상은 중복된 영상을 제외하고 128개이었다. 검색어 "장미"를 입력하였을 때 검색된 영상 URL의 수는 모두 472개였고
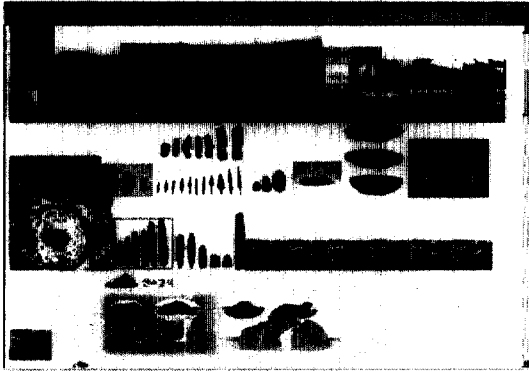
그림 11. 텍스트기반 검색 결과 (검색어 "돌")
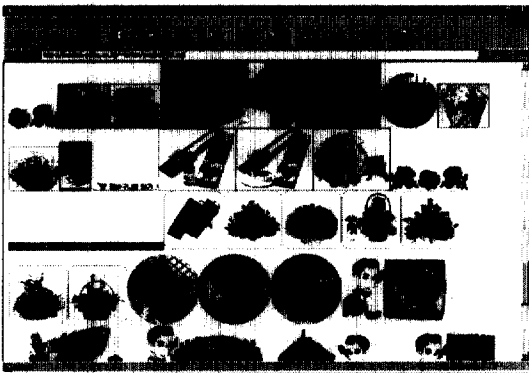
그림 12. 텍스트 기반 검색 결과(검색어:장미)

실제 수집된 영상은 중복된 영상을 제외하고 147개이었다.

이들 텍스트 기반 검색결과 출력된 영상들 중에서 사용자가 원하는 영상을 선택하여 질의 영상으로 사용한다. 그림 13은 사용자가 질의 영상으로 "돌하르방" 영상을 선택하였을 때의 내용기반 영상 검색 결과이다. 출력되는 영상 중 첫 번째 영상이 질의 영상이다. 텍스트 기반 검색을 했을 때는 서로 연관성이 없는 영상들이 많았는데 내용기반 검색을 행한 결과 질의 영상과 유사한 영상들이 유사도순으로 검색된 것을 볼 수 있다. 그림 14는 사용자가 질의 영상으로 "노란 색으로 포장된 장미 꽃다발"을 선택하였을 때의 내용기반 영상 검색 결과이다.
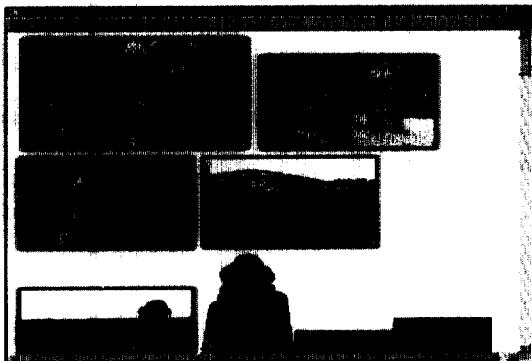
그림 13. 내용 기반 영상 검색 결과 1

그림 14. 내용 기반 영상 검색 결과 2

5. 결 론

본 논문에서 제안한 RDWIRS 는 분산 PC 클러스터를 이용함으로써 단일 컴퓨터를 이용하여 웹 영상 검색을 하였을 때 보다 많은 영상들을 보다 빠른 시간에 정확하게 검색할 수 있었다.

웹 영상의 수집 과정은 네트워의 흐름에 의존적이어서 흐름이 원활할 때와 혼잡할 때의 결과가 상당한 차이를 보였다.

단일 컴퓨터를 이용하였을 경우, 1차 문자기반 검색결과 추출된 영상 URL의 수가 500개 이상일 때 실제 수집된 영상의 수는 약 100여개 가량이었고 클러스터를 이용하였을 때 수집된 영상의 수는 약 300여개 가량이었다.

또한 처리 시간은 100개의 영상을 기준으로 했을 때 단일 컴퓨터일 경우 약 30초 가량이 걸렸으며 클러스터를 이용하였을 때 약 13초 가량이 소요되었다.

즉, 1차 검색결과 수집되는 영상의 수는 3배 정도 증가하였고 전체 처리 시간은 2배 가량 감소함으로써 실시간 처리에 더욱 근접하였다.

처리시간의 감소율이 검색 영상 개수의 증가율보다 낮은 것은 마스터 노드와 슬레이브 노드 사이의

통신을 위한 시간이 추가로 소요되었기 때문이다.

여기서 클러스터 컴퓨터의 처리시간이 약 13초인 것은 실시간이라 보기 어렵지만 클러스터의 수를 늘리면 실시간 처리가 가능할 것으로 본다.

그리고, 슬레이브 노드의 수를 3대를 이용하여 실시간 분산 웹 영상 검색을 한 결과 검색되는 영상의 수는 양적으로 증가하였지만 검색된 전체의 영상 URL 에는 미치지 못하였다.

더욱 만족할 만한 검색 결과를 얻기 위해서는 향후 영상 URL의 슬레이브 노드 분배에 있어서 영상의 크기에 따른 효율적인 분배와 클러스터 수의 확장에 따른 효율성 등에 관한 연구가 필요하다.

참 고 문 헌

[1] 장시웅, 정기동, "분산 UNIX 환경에서 Shared-Concurrent File System 의 설계 및 구현", 한국정보처리학회 논문지, 제3권 제3호, pp.617-630, May 1996.

[2] 박규석외 2, "분산 실시간 시스템을 위한 태스크 스케줄링 알고리즘의 설계 및 평가", 한국정보과학회 논문지, vol. 17 no.4. July 1990.

[3] A.P. Berman and L.G.Shapiro, "Efficient Content-Based Retrieval:Experimental Results", Proceedings of the IEEE Workshop on Conent Based Access of Image and Video Databases, pp.55-61, 1999

[4] 신순철, "리눅스 클러스터", http://medusa.linux-channel.net, 2000.6

[5] Gregory F.Pfister, "In search of clusters", Prentice Hall, pp.82-130, 1989

[6] Rajkumar Buyya, et al., "High Performance Cluster Computing", Prentice Hall, 1999

[7] 김성호, "클러스트링의 개념과 현주소", 마이크로 소프트웨어, pp.210-216, 2000.7

[8] B.Furht, S.W.Smoliar and H.Zhang, "Video and Image Processing in Multimedia Systems", Kluwer Academic Publisher, London, pp.225-270, 1995

[9] Jhon R. Smith and Shin-Fu Chang, "Automated Image Retrieval Using Colo and Texture", Columbia University Technical Report TR# 414-95-20, July, 1995

[10] Alberto del bimbo, "Visual Information Retrieval", Morgan kaufmann, pp.81-126, 1999

[11] 김희승, 영상인식, 생능출판사, pp.75-110, pp.175-199, 1993

[12] W.Y.Ma, B.S.Manjunath, Y.Luo, Y.Deng and X.Sun, "NETRA: A Content-Based Image Retrieval System", http://maya.ece.ucsb.edu/Netra, 1999

[13] W.James and Z.G.Wiederhold, O.Firschein, and S.X.Wei, "Content-based image indexing and searching using Daubechies' wavelets", International Journal of Digital Libraries, Vol.1, No.4, pp.311-328, 1998

이 은 애

1993년 서울산업대학교 전자계산학과 졸업(학사)
1996년 경희대학교 교육대학원 전자계산교육전공(석사)
2000년 경상대학교 컴퓨터과학과 박사과정 수료
1997년 3월~2001년 8월 진주 산업대학교 강사
2001년 현재 경상대학교 강사
관심분야 : 영상처리 및 검색, 신경망, 정보 보안

하 석 운

1981년 부산대학교 전자공학과 졸업(학사)
1983년 부산대학교 전자공학과 졸업(공학 석사)
1995년 부산대학교 전자공학과 졸업(공학 박사)
2001년 현재 경상대학교 컴퓨터과학과 교수, 컴퓨터 정보통신 연구원
관심분야 : 영상처리 및 검색, 컴퓨터 비전, 신경망